ИНФОРМАТИКА

2013 **июль-сентябрь** **№ 3**

УДК 681.32

А.А. Иванюк

ПРОЕКТИРОВАНИЕ КОНФИГУРИРУЕМОГО СДВИГОВОГО РЕГИСТРА С ЛИНЕЙНОЙ ОБРАТНОЙ СВЯЗЬЮ

Рассматривается методика проектирования конфигурируемого сдвигового регистра, для которого возможно задание его разрядности в различных режимах функционирования. Предложенная схема сдвигового регистра с линейной обратной связью позволяет использовать его в качестве циклического сдвигового регистра, генератора М-последовательности, счетчика Джонсона и одноканального сигнатурного анализатора. Приводится оценка аппаратурных затрат на реализацию конфигурируемого сдвигового регистра.

Введение

Среди многообразия цифровых генераторов псевдослучайных последовательностей (ГПП) наиболее широкое распространение получили генераторы М-последовательностей, построенные на основе сдвиговых регистров с линейной обратной связью (от англ. Linear Feedback Shift Register, LFSR) [1]. В задачах контроля и диагностики средств вычислительной техники LFSR используются для синтеза генераторов псевдослучайных тестовых последовательностей и сигнатурных анализаторов [2]; в криптографии – для генерирования символов псевдослучайных числовых последовательностей с дискретным равномерным распределением [3], синтеза схем шифрования в потоковых криптосистемах [4]; в системах телекоммуникаций – для аппаратной реализации схем помехоустойчивого кодирования [5], схем скремблирования [6] и т. д.

В статье рассматривается один из возможных вариантов реализации конфигурируемого LFSR, для которого предусмотрено динамическое изменение его разрядности и режимов функционирования.

1. Теоретические основы проектирования цифровых устройств на основе LFSR

В основе структуры LFSR лежит *n*-разрядный сдвиговый регистр, проектируемый, как правило, при помощи синхронных *D*-триггеров. Синтез LFSR осуществляется на основе характеристического полинома

$$\varphi(x) = 1 \oplus \alpha_1 x^1 \oplus \alpha_2 x^2 \oplus \ldots \oplus \alpha_{n-1} x^{n-1} \oplus \alpha_n x^n, \quad (1)$$

где $n = \deg(\varphi(x))$ определяет число разрядов LFSR, а коэффициенты $\alpha_i \in \{0,1\}$ $(i = \overline{1,n})$ используются для формирования значения сигнала в цепи обратной связи.

Пусть $d_i \in \{0,1\}$ $(i = \overline{1,n})$ есть значение, хранящееся на *i*-м триггере, а $D^{(k)} = \{d_1^{(k)}, d_2^{(k)}, \ldots, d_n^{(k)}\}$ – *n*-разрядное двоичное слово, являющееся состоянием LFSR в *k*-й такт функционирования. Предположим, что в (*k*+1)-й такт функционирования при наступлении фронта сигнала синхронизации, являющегося общим для всех триггеров LFSR, осуществляется операция поразрядного сдвига двоичного слова, такая, что

$$d_i^{(k+1)} = d_{i-1}^{(k)}, \forall i = \overline{2,n}. \quad (2)$$

Новое значение младшего разряда LFSR при этом вычисляется исходя из значений коэффициентов полинома (1):

$$d_1^{(k+1)} = \bigoplus_{i=1}^{n} \alpha_i d_i^{(k)}. \quad (3)$$

Соответствующий выбор полинома (1) и начального состояния $D^{(0)}$ определяет вид последовательности, вырабатываемой LFSR. Например, при $\varphi(x)=1\oplus x^n$ и $D^{(0)}=\{1,0,0,...,0\}$ последовательность вырабатываемых двоичных слов $(D^{(0)}, D^{(1)}, D^{(2)}, ..., D^{(n-1)}, D^{(0)})$ будет представлять собой циклическую двоичную последовательность типа «one hot» с периодом повторения символов, равным n. С учетом того что $\alpha_1=\alpha_2=...=\alpha_{n-1}=0$ и $\alpha_n=1$, выражение (3) принимает следующий вид:

$$d_1^{(k+1)}=d_n^{(k)}. \tag{4}$$

Выражение (4) совместно с (2) может быть использовано для синтеза структуры генератора вышеописанной циклической последовательности.

Если характеристический полином $\varphi(x)$ является примитивным, период повторения вырабатываемых символов равен 2^n-1. Такого рода двоичные последовательности называются М-последовательностями [2] и по своим вероятностным характеристикам являются псевдослучайными последовательностями, при этом LFSR называется генератором М-последовательности либо ГПП.

Для синтеза *n*-разрядного ГПП на основе LFSR необходимо выбрать соответствующий примитивный полином степени *n* ($\deg(\varphi(x))=n$). Известно, что число примитивных полиномов степени *n* над полем GF(2) можно вычислить по формуле [7]

$$M(n)=\frac{L(2^n-1)}{n}, \tag{5}$$

где L – функция Эйлера.

Например, для $n=4$ существует $M(4)=2$ примитивных полинома $\varphi_1(x)=1\oplus x\oplus x^4$ и $\varphi_2(x)=1\oplus x^3\oplus x^4$, на основе которых можно спроектировать два генератора М-последовательности. С учетом одинаковых начальных состояний такие генераторы за 15 тактов функционирования выработают 15 символов двух М-последовательностей, но с различным порядком их следования. Это свойство характерно для всех ГПП, синтезированных на основе примитивных полиномов с одинаковым значением их старших степеней.

Генераторы М-последовательностей часто применяются в качестве источников тестовых воздействий при решении задач тестирования цифровых устройств, при которых реакции на тестовые воздействия сжимаются в компактную характеристику, называемую сигнатурой [2]. Аппаратура сжатия при этом называется сигнатурным анализатором [2], структура которого может быть синтезирована по схожим принципам, что и генератор М-последовательности. В общем случае одноканальный сигнатурный анализатор (ОСА) представляет собой ГПП, в цепи обратной связи которого присутствует дополнительный элемент XOR (исключающее ИЛИ), на один из входов которого подается символ сжимаемой последовательности. При этом значение младшего разряда LFSR описывается следующим образом:

$$d_1^{(k+1)}=d_0^{(k)}\oplus\left(\bigoplus_{i=1}^{n}\alpha_i d_i^{(k)}\right), \tag{6}$$

где $d_0^{(k)}\in\{0,1\}$ – значение сжимаемого символа.

При условиях, что $D^{(0)}\neq\{0,0,0,...,0\}$, $d_0^{(k)}=0$ $(\forall k=0,1,2,...)$ и α_i есть коэффициенты примитивного полинома $\varphi(x)$, аппаратура ОСА будет функционировать в качестве генератора М-последовательности. Если $\varphi(x)=1\oplus x^n$ и $d_0^{(k)}=1$ $(\forall k=0,1,2,...)$, выражение (6) принимает следующий вид:

$$d_1^{(k+1)} = 1 \oplus d_n^{(k)} = \overline{d_n^{(k)}}, \tag{7}$$

что для случая $n = 2^r$ $(\forall r = 1, 2, 3, ...)$ является выражением для вычисления нового значения младшего разряда счетчика Джонсона [8], вырабатывающего псевдослучайную последовательность с периодом, равным $2n$. Таким образом, соответствующие значения α_i и $d_0^{(k)}$ при заданном n позволяют посредством выражений (2) и (6) описать функционирование четырех различных цифровых устройств: генератора циклической последовательности, генератора М-последовательности, одноканального сигнатурного анализатора и счетчика Джонсона.

С целью определения произвольной разрядности в пределах значения n вышеперечисленных аппаратных структур введем дополнительные коэффициенты $\beta_j \in \{0,1\}$ $(\forall j = \overline{1, n})$ (рис. 1).

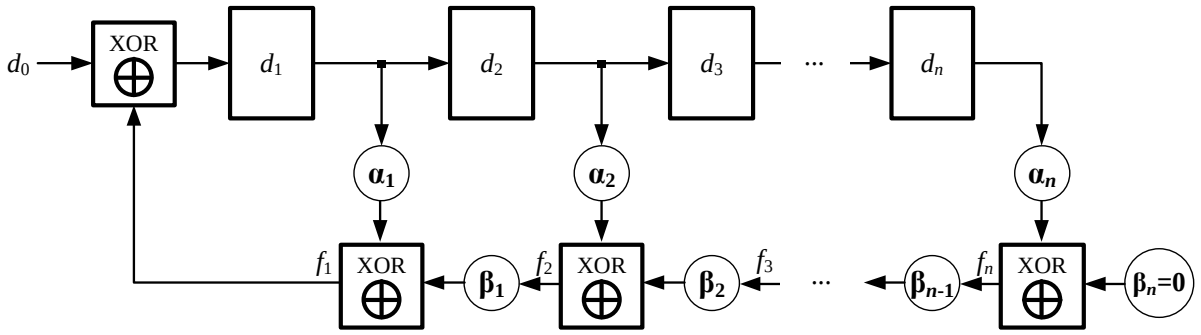


Рис. 1. Обобщенная структура конфигурируемого сдвигового регистра

Нулевое значение коэффициента β_j означает, что соответствующий ему разряд d_j является старшим разрядом в конфигурируемой структуре и все последующие разряды d_l $(j < l \le n)$ не участвуют в формировании значения сигнала f_1 в цепи обратной связи. При этом значение f_1 можно выразить следующим образом:

$$\begin{aligned} f_1 &= \alpha_1 d_1 \oplus \beta_1 f_2 = \alpha_1 d_1 \oplus \alpha_2 \beta_1 d_2 \oplus \beta_1 \beta_2 f_3 = ... = \\ &= \alpha_1 d_1 \oplus \alpha_2 \beta_1 d_2 \oplus \alpha_3 \beta_1 \beta_2 d_3 \oplus ... \oplus \alpha_n \beta_1 \beta_2 ... \beta_{n-1} f_n. \end{aligned} \tag{8}$$

С учетом того что для старшего используемого разряда $f_n = \alpha_n d_n$ $(\beta_n = 0)$, выражение (8) можно записать в более компактной форме:

$$f_1 = \alpha_1 d_1 \oplus \left(\bigoplus_{i=1}^{n-1} \alpha_{i+1} d_{i+1} \prod_{j=1}^{i} \beta_j \right). \tag{9}$$

Таким образом, единичные значения коэффициентов $\beta_1 = \beta_2 = ... = \beta_m = 1$ определяют количество разрядов $m \le n$, участвующих в конфигурации сдвигового регистра с линейной обратной связью.

В общем случае значение младшего разряда конфигурируемого сдвигового регистра описывается выражением

$$d_1^{(k+1)} = d_0^{(k)} \oplus \alpha_1 d_1^{(k)} \oplus \left(\bigoplus_{i=1}^{n-1} \alpha_{i+1} d_{i+1}^{(k)} \prod_{j=1}^{i} \beta_j \right). \tag{10}$$

Развитие идеи компактного тестирования привело к появлению конфигурируемых структур на подобие BILBO (от англ. Built-In Logic Block Observer) [9], которые, будучи построенными на LFSR, могут функционировать как в качестве генераторов тестовых последовательностей, так и в качестве сигнатурных анализаторов. Двойственное функционирование BILBO обусловлено наличием реконфигурируемых блоков, которые обеспечивают соответствующую коммутацию сигналов в зависимости от задаваемого режима. Для BILBO определены четыре основных режима: режим нормального функционирования, при котором триггеры, входящие в состав BILBO, играют роль элементов памяти устройства; режим сдвигового регистра; режим генератора тестовых последовательностей (ГТП) и режим сигнатурного анализатора (СА) [9].

В работе [10] было показано, что для увеличения достоверности встроенного самотестирования посредством BILBO необходимо применять ГТП и СА с использованием различных полиномов.

В общем случае задачу проектирования конфигурируемого LFSR можно сформулировать как задачу синтеза сдвигового регистра с различными задаваемыми коэффициентами α_i и с различным значением числа разрядов в пределах n.

Для решения данной задачи было предложено множество подходов [10–14]. Так, в работе [10] предлагается структура LFSR с фиксированным параметром n и возможностью задания различных коэффициентов α_i. В работе [11] предлагается 128-разрядный сдвиговый регистр с множеством фиксированных коэффициентов α_i для возможности реализации LFSR произвольной разрядности в пределах от $n = 8$ до $n = 128$. Дальнейшим развитием работы [11] стала публикация [12], предлагающая идею реконфигурируемого LFSR с целью обеспечения различных базовых операций для программно-определяемых радиосистем (от англ. Software-Defined Radio, SDR). В работе [13] была предложена архитектура LFSR, состоящая из 64 базовых реконфигурируемых элементов, каждый из которых содержит настраиваемый 8-разрядный сдвиговый регистр данных и 32 8-разрядных конфигурационных регистра, позволяющих настраивать структуру LFSR на произвольную разрядность для осуществления различных операций над элементами полей GF(2), GF(2^8), GF(2^{16}) либо GF(2^{32}). В работе [14] рассматривается задача аппаратной реализации генераторов псевдослучайных последовательностей с перестраиваемой структурой на основе теории клеточных автоматов.

В настоящей работе рассмотрим методику проектирования конфигурируемого сдвигового регистра посредством языка VHDL с дальнейшей его реализацией для программируемых логических интегральных схем типа FPGA.

2. Проектирование конфигурируемого сдвигового регистра

Модульность и высокий уровень абстракции языка VHDL позволяют описывать схемотехнические элементы цифровых устройств произвольной сложности [15]. Кроме того, VHDL позволяет составлять параметризированные проектные описания для случая итерационных цифровых структур. В связи с этим для составления VHDL-описаний LFSR-структур могут быть применены следующие подходы:

1. Составление непараметризированного описания с фиксированными значениями n и α_i.

2. Составление параметризированного описания с произвольно задаваемыми значениями n и α_i.

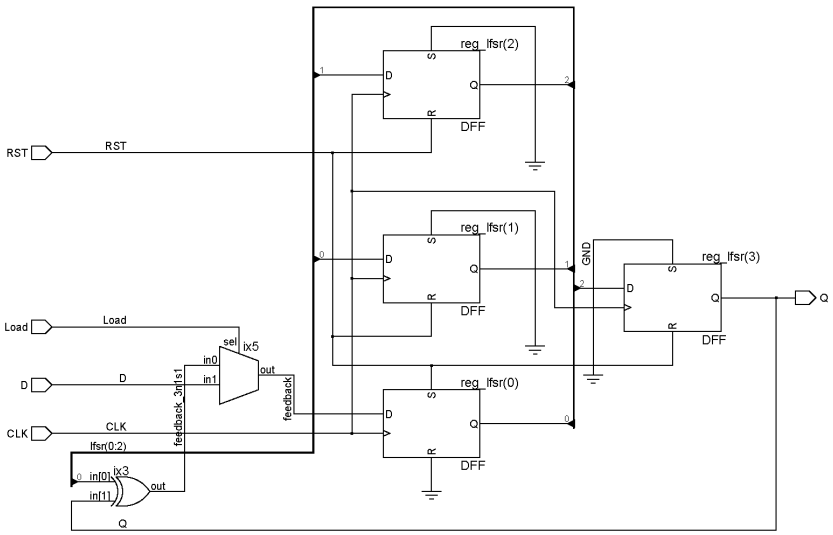
3. Составление параметризированного описания конфигурируемого сдвигового регистра с произвольно задаваемыми значениями α_i и β_j.

Применение первого подхода позволяет достичь минимальных аппаратурных затрат при синтезе описываемой LFSR-структуры, однако изменение разрядности LFSR либо множества коэффициентов α_i приведет к изменению исходного VHDL-описания и повторному циклу проектирования устройства.

Рассмотрим пример непараметризированного описания генератора М-последовательности и результат его синтеза для $n = 4$ и $\varphi(x) = 1 \oplus x \oplus x^4$ (рис. 2).

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity LFSR4 is
   port ( CLK, RST, Load, D : in   std_logic;
                              Q : out std_logic );
end LFSR4;

architecture Behavioral of LFSR4 is
signal feedback   : std_logic;
signal lfsr       : std_logic_vector( 0 to 3 );
constant init     : std_logic_vector( 0 to 3 ) := "1000";
begin
PSHIFT: process( RST, CLK, feedback, lfsr )
begin
        if ( RST = '1' ) then
                lfsr <= init;
        elsif rising_edge( CLK ) then
                lfsr <= feedback & lfsr( 0 to 2);
        end if;
end process;

PFB: process( Load, D, lfsr )
begin
        if ( Load = '1' ) then
                feedback  <= D;
        else
                feedback <= lfsr( 0 ) xor lfsr( 3 );
        end if;
end process;

Q <= lfsr(3);

end behavioral;
```

а)



б)

Рис. 2. Генератор псевдослучайной последовательности:
а) описание ГПП для $n = 4$; *б)* результат его RTL-синтеза

Из рис. 2 видно, что параметр $(n-1)$ в явном виде присутствует как при объявлении сигналов, так и при описании подсхем LFSR, а именно в процессе PSHIFT, описывающем синхронный сдвиговый регистр; в процессе PFB, описывающем двухвходовый элемент XOR и мультиплексор, необходимые для формирования значения сигнала линейной обратной связи, и в последнем параллельном операторе, описывающем передачу одного бита вырабатываемой последовательности на выходной порт устройства.

Определение начального состояния ГПП возможно двумя способами: асинхронно при установке единичного значения сигнала на входном порту *RST*, при этом сдвиговый регистр принимает фиксированное значение "1000"; синхронно при удержании единичного значения сигнала на входном порту *Load*. В первом случае изменение инициализирующего значения регистра возможно только при составлении его проектного описания, во втором случае – во время функционирования устройства по назначению.

Основной задачей при трансформации представленного описания в параметризированное описание ГПП для целочисленного параметра *n* будет являться описание оператора, формирующего значение сигнала обратной связи в процессе PFB для произвольно задаваемых коэффициентов α_i. Для этого введем generic-параметр *ALPHA* безразмерного типа std_logic_vector, значение которого будет определять бинарную маску множества коэффициентов $\{\alpha_1, \alpha_2, ..., \alpha_n\}$ (рис. 3).

Из рис. 3 видно, что параметр $ALPHA := "1011"$ задает порождающий полином $\varphi(x) = 1 \oplus x^3 \oplus x^4$. Размерность таких объектов, как lfsr и init, можно определить посредством оператора **range**, а значение номера старшего разряда lfsr – атрибута оператора **high**. Вычисление значения сигнала в линии обратной связи можно произвести посредством оператора **for** в процессе PFB.

Представленный пример параметризированного описания является синтезируемым, и при значении $ALPHA := "1001"$ результатом RTL-синтеза является схема, изображенная на рис. 2. В общем случае для возможности использования различных ГПП в HDL-описаниях проектировщику достаточно определить компоненту LFSRn с указанием одного параметра ALPHA.

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity LFSRn is
   generic (
         ALPHA : std_logic_vector := "1011" );
   port ( CLK     : in  std_logic;
         RST     : in  std_logic;
         Load    : in  std_logic;
         D       : in  std_logic;
         Q       : out std_logic );
end LFSRn;

architecture behavioral of LFSRn is
signal feedback : std_logic;
signal lfsr     : std_logic_vector( ALPHA'range );
begin

PSHIFT: process( CLK, RST, lfsr )
variable init : std_logic_vector( ALPHA'range );
begin
         init    := ( others => '0' );
         init( 0 ) := '1';
         if ( RST = '1' ) then
                  lfsr <= init;
         elsif rising_edge( CLK ) then
                  lfsr <= feedback & lfsr( 0 to ALPHA'high-1 );
         end if;
end process;
```

```
PFB: process( Load, D, lfsr )
variable fb : std_logic;
begin
         fb := '0';
         if ( Load = '1' ) then
                  feedback  <= D;
         else
                  for i in ALPHA'range loop
                           if ( ALPHA( i ) = '1' ) then
                                    fb := fb xor lfsr( i );
                           end if;
                  end loop;
                  feedback <= fb;
         end if;
end process;

Q <= lfsr( ALPHA'high );

end behavioral;
```

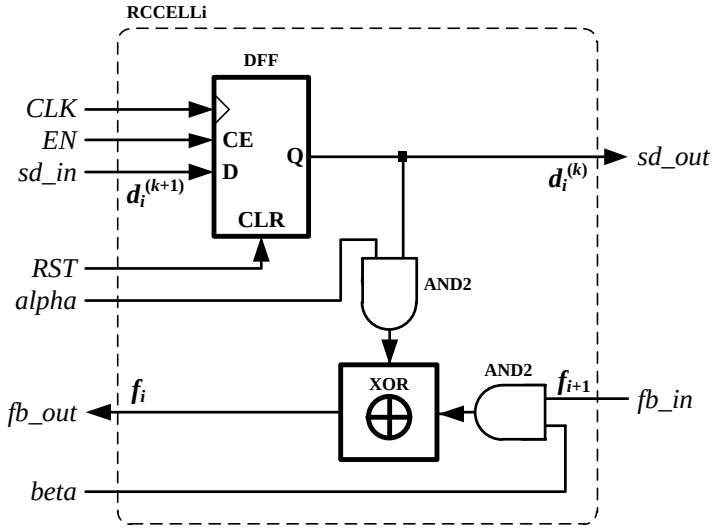
Рис. 3. Параметризированное описание ГПП

Структура конфигурируемого сдвигового регистра должна иметь возможность определять его разрядность в пределах *n* и значения коэффициентов α_i непосредственно в процессе функционирования. Для этого согласно выражению (10) модифицируем разряды LFSR, представленные *D*-триггерами, следующим образом. Для каждого разряда LFSR общими сигналами будут: *CLK* – сигнал синхронизации, *RST* – сигнал асинхронной инициализации, *EN* – асинхронный сигнал разрешения. Каждый разряд LFSR должен иметь входной порт данных *sd_in* и выходной порт данных *sd_out* для возможности обеспечения микрооперации сдвига. Для каждого разряда LFSR введем два дополнительных входных порта: *alpha* – для передачи значения коэффициента α_i и *beta* – для передачи значения сигнала, определяющего старший разряд сдвигового регистра. Для этого введем в структуру каждого элемента дополнительную аппаратуру, которая будет отвечать за конфигурацию цепи обратной связи, что потребует наличия двух дополнительных портов: входного порта сигнала обратной связи *fb_in* и выходного *fb_out*.

С учетом описанных модификаций структурная схема одного разряда конфигурируемого сдвигового регистра может выглядеть следующим образом (рис. 4).

Конфигурация разряда RCCELLi осуществляется посредством задания значений сигналов *alpha* и *beta*.

Например, в случае $alpha = '0'$ и $beta = '0'$ разряд играет роль элемента памяти и может быть использован для конфигурации линейного сдвигового регистра без обратной связи. При условии $alpha = '0'$ и $beta = '1'$ значение, хранимое на триггере, не участвует в формировании обратной связи, а значение сигнала на входе *fb_in* транслируется на выходной порт *fb_out*. Условие $alpha = '1'$ и $beta = '0'$ может быть использовано для конфигурации старшего разряда LFSR, значение которого непосредственно передается по линии обратной связи. Четвертое условие $alpha = '1'$ и $beta = '1'$ конфигурирует разряд LFSR как разряд, значение которого участвует в формировании сигнала обратной связи.




```
...
PDFF: process( CLK, RST, EN, sd_in )
begin
        if ( RST = '1' ) then
                dff <= '0';
        elsif rising_edge( CLK ) then
                if ( EN = '1' ) then
                        dff <= sd_in;
                end if;
        end if;
end process;

sd_out <= dff;

and0  <= dff    when ( alpha = '1' ) else '0';
and1  <= fb_in when ( beta  = '1' ) else '0';

fb_out <= and0  xor and1;
...
```

б)

Рис. 4. Конфигурируемый LFSR:
а) структура одного разряда; *б)* VHDL-описание

На основе имеющейся компоненты RCCELLi можно сформировать структуру *n*-разрядного конфигурируемого сдвигового регистра (рис. 5).

Младший разряд RCCELL0 дополнен двухвходовым мультиплексором, двумя элементами XOR и одним элементом AND для возможности изменения значения сигнала обратной связи f_1. Дополнительные схемы мультиплексора и двух демультиплексоров необходимы для управления процессом последовательной загрузки инициализирующих данных в триггеры элементов RCCELLi либо в память конфигурации в зависимости от значения сигнала на входе *ADR*.

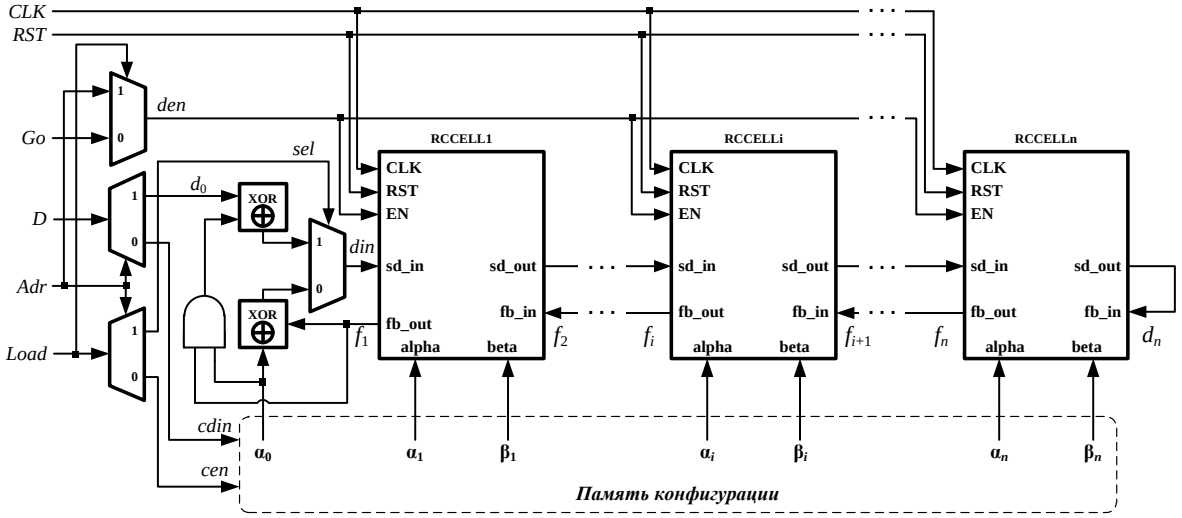


Рис. 5. Структура *n*-разрядного конфигурируемого сдвигового регистра

Так, последовательная загрузка данных из порта *D* в триггеры LFSR ($din = D$) осуществляется при выполнении следующих условий: $Adr = '1'$, $Load = '1'$ и $\alpha_0 = '0'$. Полный перечень возможных конфигураций представленной схемы приведен в табл. 1.

Таблица 1

Возможные варианты конфигураций

Adr	*Load*	*Go*	α_0	*den*	*cen*	*sel*	*din*	*cdin*	Пояснение
X	0	0	X	0	0	0	X	X	режим регистра хранения
X	0	1	0	1	0	0	f_1	X	режим ГПП сдвигового регистра
X	0	1	1	1	0	0	$\overline{f_1}$	X	режим ГПП счетчика Джонсона
0	1	X	X	0	1	0	X	*D*	режим инициализации памяти конфигурации
1	1	X	0	1	0	1	d_0	0	режим инициализации LFSR
1	1	X	1	1	0	1	$d_0 \oplus f_1$	0	режим одноканального сигнатурного анализатора

Память конфигурации, хранящая значения коэффициентов $\{\alpha_0, ..., \alpha_n\}$ и $\{\beta_1, ..., \beta_n\}$, может представлять собой набор из двух сдвиговых регистров, значения которых, как и значения LFSR, могут загружаться из общего входного порта *D*. Предположим, что значения $\{\alpha_0, ..., \alpha_n\}$ хранятся в $(n+1)$-разрядном сдвиговом регистре reg_alpha, а $\{\beta_1, ..., \beta_n\}$ – в n-разрядном сдвиговом регистре reg_beta. Покажем, что для определения значения reg_beta достаточно последовательного определения значений разрядов регистра reg_alpha. Во время процедуры инициализации ($RST = '1'$) оба регистра принимают нулевые значения. После инициализации значения коэффициентов порождающего полинома $\varphi(x)$ ($\deg(\varphi(x)) = k$) последовательно записываются в регистр reg_alpha, начиная со старшего значимого $\alpha_k = '1'$. После k тактов сигнала синхронизации в регистр записывается значение младшего коэффициента $\alpha_0 = '0'$ для возможности конфигурации структуры ГПП. Таким образом, по прошествии $k+1$ тактов синхронизации содержимое регистра reg_alpha равно $\{0, \alpha_1, \alpha_2, ..., \alpha_{k-1}, 1, 0, ..., 0\}$, что эквивалентно порождающему полиному вида $\varphi(x) = 1 \oplus \alpha_1 x \oplus \alpha_2 x^2 \oplus ... \oplus \alpha_{k-1} x^{k-1} \oplus x^k$. Для корректной конфигурации структуры LFSR значение регистра reg_beta должно принимать следующее значение: $\{\beta_1, ..., \beta_{k-1}, \beta_k, \beta_{k+1}, ..., \beta_n\} = \{1, ..., 1, 0, 0, ..., 0\}$. Видно, что единичные значения $k-1$ младших разрядов регистра reg_beta могут устанавливаться параллельно с приемом в регистр reg_alpha очередного значения коэффициента α_i без учета $\alpha_k = '1'$, которое может служить индикатором начала процесса заполнения регистра reg_beta.

Исходя из сказанного выше, память конфигурации можно спроектировать в виде схемы, изображенной на рис. 6.

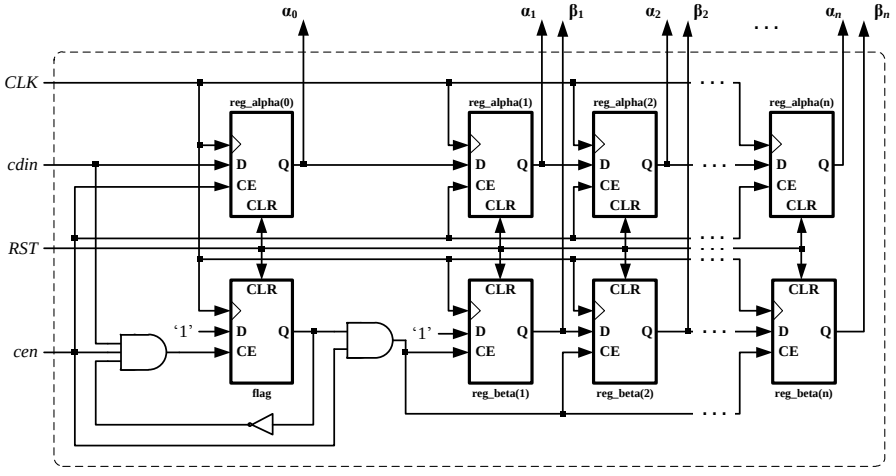


Рис. 6. Структура памяти конфигурации

В табл. 2 приведены некоторые значения регистров reg_alpha и reg_beta и соответствующие конфигурации предложенной схемы LFSR для $n=8$.

Таблица 2

Варианты конфигурирования LFSR для $n=8$

Регистр	Номер разряда регистра									Пояснения	
	0	1	2	3	4	5	6	7	8		
reg_alpha	0	0	0	0	0	0	0	0	1	8-разрядный циклический сдвиговый регистр	$d_1^{(k+1)}=d_8^{(k)}$
reg_beta	-	1	1	1	1	1	1	1	0		
reg_alpha	1	0	0	0	1	0	0	0	0	4-разрядный счетчик Джонсона	$d_1^{(k+1)}=\overline{d_4^{(k)}}$
reg_beta	-	1	1	1	0	0	0	0	0		
reg_alpha	0	1	0	0	0	1	1	0	1	8-разрядный ГПП, $\varphi(x)=1\oplus x\oplus x^5\oplus x^6\oplus x^8$	$d_1^{(k+1)}=d_1^{(k)}\oplus d_5^{(k)}\oplus d_6^{(k)}\oplus d_8^{(k)}$
reg_beta	-	1	1	1	1	1	1	1	0		
reg_alpha	1	1	0	0	0	0	1	0	0	6-разрядный ОСА, $\varphi(x)=1\oplus x\oplus x^6$	$d_1^{(k+1)}=d_0^{(k)}\oplus d_1^{(k)}\oplus d_6^{(k)}$
reg_beta	-	1	1	1	1	1	0	0	0		

Для определения конфигурации представленного LFSR необходимо выполнить следующую последовательность действий.

1. Осуществить инициализацию LFSR посредством подачи на вход *RST* сигнала с единичным логическим значением.

2. Задать начальное состояние регистра данных путем установки следующих значений на входных портах: $Adr='1'$, $Load='1'$, для каждого фронта сигнала синхронизации на входе *CLK* формировать бит инициализации на входном порту *D*. Процесс инициализации может занимать от 1 до n периодов сигнала синхронизации.

3. Задать значение памяти конфигурации путем установки следующих значений на входных портах: $Adr='0'$, $Load='1'$, для каждого фронта сигнала синхронизации на входе *CLK* формировать бит конфигурации на входном порту *D*. Процесс конфигурации потребует $n+1$ периодов сигнала синхронизации.

4. В зависимости от заданного значения конфигурации (см. табл. 1) обеспечить функционирование LSFR путем установки соответствующих сигналов на входных портах *Go*, *ADR* и *Load*.

Представленную структуру конфигурируемого LFSR можно описать, используя смешанный стиль языка VHDL, который подразумевает использование структурного и поведенческого подмножества языка в одном проекте.

3. Анализ результатов аппаратной реализации

Функциональное моделирование составленного VHDL-описания конфигурируемого LFSR производилось при помощи программного средства ISim Simulator, входящего в состав пакета проектирования цифровых устройств Xilinx ISE [16]. Процесс технологического синтеза для различных параметров *n* осуществлялся для кристаллов FPGA Xilinx Spartan-3E XC3S500E [17]. С точки зрения RTL-представления схемная реализация конфигурируемого LFSR разрядности *n* требует наличия следующих компонентов: $3n+2$ триггеров *D*-типа, $n+2$ двухвходовых элементов XOR, $2n+4$ двухвходовых элементов AND, двух мультиплексоров с конфигурацией 2×1 и двух демультиплексоров с конфигурацией 1×2.

Интегральная схема XC3S500E имеет в своем составе 1164 CLB-блока, каждый из которых содержит четыре Slice-блока: два SliceM- и два SliceL-блока. В свою очередь, каждый из перечисленных Slice-блоков состоит из двух генераторов функций LUT, способных реализовывать произвольную переключательную функцию от четырех переменных, и из двух элементов памяти, которые могут быть настроены в качестве синхронных триггеров D-типа. Таким образом, для выбранного кристалла FPGA имеется 4656 Slice-блоков, содержащих 9312 триггеров (DFF) и 9312 LUT-блоков.

Оценка аппаратурных затрат на реализацию конфигурируемого LFSR разрядности *n* изображена на рис. 7.

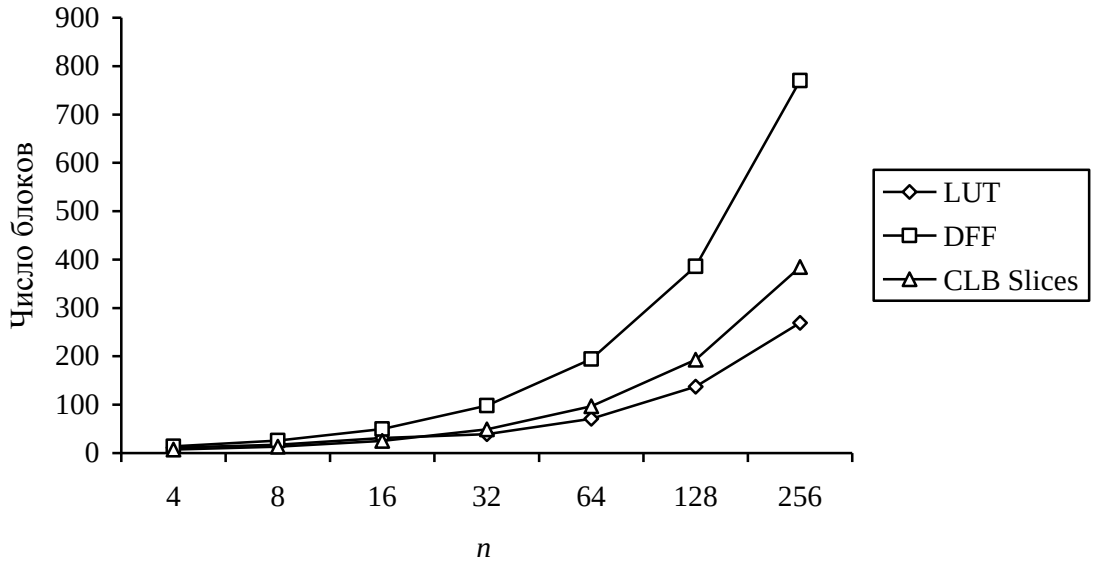


Рис. 7. График зависимости числа технологических блоков от параметра *n*

Так, при значении $n = 256$ для реализации конфигурируемого LFSR потребуется 385 Slice-блоков либо 269 LUT-блоков и 770 триггеров, что в совокупности составляет 8,27 % от общих ресурсов кристалла XC3S500E. В случае реализации 256-разрядного ГПП, представленного параметризированным описанием (см. рис. 4), аппаратурные затраты составят 256 триггеров и 2 LUT-блока либо 147 Slice-блоков, что в 2,6 раза меньше по сравнению с конфигурируемым LFSR. Однако в отличие от стандартной схемной реализации предложенный конфигурируемый LFSR способен изменять свое функционирование путем задания нового значения памяти конфигурации непосредственно в рабочем режиме.

Представленная методика проектирования конфигурируемого генератора может быть применена для синтеза LFSR с внутренними сумматорами по модулю два и схем многоканальных сигнатурных анализаторов.

Заключение

В статье предложена схемная реализация цифрового конфигурируемого генератора псевдослучайных последовательностей, позволяющая реализовывать различные режимы функционирования регистра хранения, сдвигового регистра, счетчика Джонсона, генератора М-последовательности, одноканального сигнатурного анализатора. Задание конкретного режима осуществляется путем последовательной инициализации памяти конфигурации представленной схемы. Конфигурируемый генератор может быть применен в приложениях, требующих использования различных псевдослучайных последовательностей, например при реализации средств встроенного самотестирования цифровых устройств.

Список литературы

1. VLSI Test Principles and Architecture / L.-T. Wang [et al.]. – San Francisco : Morgan Kaufman, 2006. – 777 p.

2. Ярмолик, В.Н. Контроль и диагностика цифровых узлов ЭВМ / В.Н. Ярмолик. – Минск : Наука и техника, 1988. – 240 с.

3. Gentle, J.E. Random Number Generation and Monte Carlo Methods / J.E. Gentle. – N. Y. : Springer-Verlag, 2003. – 281 p.

4. Stream Ciphers and Number Theory / T.W. Cusick [et al.]. – Amsterdam : Elsevier, 2004. – 413 p.

5. Moon, T.K. Error Correction Coding: Mathematical Methods and Algorithms / T.K. Moon. – New Jersey : John Wiley & Sons, 2005. – 756 p.

6. Wesolowski, K. Introduction to Digital Communication Systems / K. Wesolowski. – Chichester : John Wiley & Sons, 2009. – 561 p.

7. Shparlinski, I.E. Finite Fields: Theory and Computation / I.E. Shparlinski. – Dordrecht : Kluwer Academic Publishers, 1999. – 525 p.

8. Уэйкерли, Дж. Проектирование цифровых устройств: в 2 т. / Дж. Уэйкерли. – М. : Постмаркет, 2002. – Т. 2. – 528 с.

9. Konemann, B. Built-In Logic Block Observation Techniques / B. Konemann, J. Mucha, G. Zwiehoff // International Test Conference (ITC'79) : Proc. on IEEE Int. Conf. – New Jersey, USA, 1979. – P. 37–42.

10. Bolling, R. Reconfigurable Linear Feedback Register Design, Analysis and Applications / R. Bolling, S.A. Al-Arian // Circuits and Systems (ISCAS'94) : Proc. on IEEE Int. Symp. – London, England, UK, 1994. – Vol. 4. – P. 87–90.

11. A Reconfigurable Linear Feedback Shift Register (LFSR) for the Bluetooth System / P. Kitos [et al.] // Electronics, Circuits and Systems (ICECS'01) : Proc. On IEEE Int. Conf. – Malta, 2001. – P. 991–994.

12. Alaus, L. A Reconfigurable Linear FeedBack Shift Register Operator for Software Defined Radio Terminal / L. Alaus, D. Noguet, J. Palicot // Wireless Pervasive Computing (ISWPC'2008) : Proc. of Int. Symp. – Santorini, Greece, 2008. – P. 319–323.

13. Zhiyuan, W. A Kind of Reconfigurable Linear Feedback Register Design / W. Zhiyuan, H. Jianhua, G. Ziming // Information Technology and Applications (IFITA'2009) : Proc. of Int. Forum. – Chengdu, China, 2009. – P. 657–660.

14. Мурашко, И.А. Автоматизированное проектирование генераторов псевдослучайных последовательностей с использованием аппарата клеточных автоматов / И.А. Мурашко, Д.Е. Храбров // Информационные технологии и системы 2012 (ИТС 2012) : материалы Междунар. науч. конф., Минск, Беларусь, 24 октября 2012 г. – Минск : БГУИР, 2012. – С. 188–189.

15. Chu, P.P. RTL Hardware Design Using VHDL / P.P. Chu. – New Jersey : John Wiley & Sons, 2006. – 669 p.

16. ISE Design Suite: Logic Edition [Electronic resource]. – Xilinx Inc., 2012. – Mode of access : http://www.xilinx.com/products/design-tools/ise-design-suite/logic-edition.htm. – Date of access : 28.12.2012.

17. Spartan-3E FPGA Family Data Sheet [Electronic resource]. – Xilinx Inc., 2006. – Mode of access : http://www.xilinx.com/support/documentation/data_sheets/ds312.pdf. – Date of access : 28.12.2012.

Поступила 18.01.13

Белорусский государственный университет информатики и радиоэлектроники,
Минск, ул. П. Бровки, 6
e-mail: ivaniuk@bsuir.by

A.A. Ivaniuk

DESIGNING CONFIGURABLE SHIFT REGISTER WITH A LINEAR FEEDBACK

A method for designing a configurable shift register is considered, which allows setting its capacity in various operation modes. The suggested shift register with a linear feedback can be used as a cyclic shift register, a generator of M-sequences, Johnson's counter and a single-channel signature analyzer. An assessment of the relevant hardware implementation expenses is given.